A Semantic-Driven Recommendation System using Topic modeling

Syed Mehlial Hassan Kazmi
Georgia Institute of Technology
skazmi34@gatech.edu

Shilpa G
Georgia Institute of Technology
sg306@gatech.edu

Abstract

In the realm of e-commerce, the ability to provide personalized recommendations based on user reviews is a key to enhancing user experience. This project aims to develop a semantic-driven recommendation system by performing topic modeling on Amazon Fine Food Reviews. The traditional recommendation systems often overlook the rich semantic information available in user reviews. This project aims to leverage this information to not only provide more personalized recommendations but also understand why certain recommendations are made. The project combines Natural Language Processing (NLP) and recommendation systems, two significant areas in machine learning, to solve a real-world problem.

1. Introduction

With the advent of the e-commerce platforms such as Amazon, Alibaba and JD.com the world of retail has changed forever, digital economy is far more substantial than physical economy. The success of these giants greatly depend on recommendation systems they have developed over the past years. These recommendation systems increase customer engagement, give personalized feedback, contribute to revenue growth by up-selling and cross-selling and mitigate the information overload as user can get easily lost in the vast catalog of products. Discovering and developing improved recommendation systems is crucial to survival of these businesses.

Recommender Systems (RS) can be classified into three broad categories Content based, Collaborative based and hybrid based. Content-based filtering (CBF) utilizes user's information (gender, age, location, interactions on social media, etc.) to forecast their preferences without taking into account any information about other users [2]. The technique of collaborative Filtering (CF) recommends items, to a target user, based on the opinions and interactions of other users [1, 6]. We should point out that CF approach has some major advantages over CBF since it can be applied on a context that does not contain much information associated with the user/item. Hybrid filtering approach combines two or more filtering techniques in order to exploit merits of each one of these techniques [4].

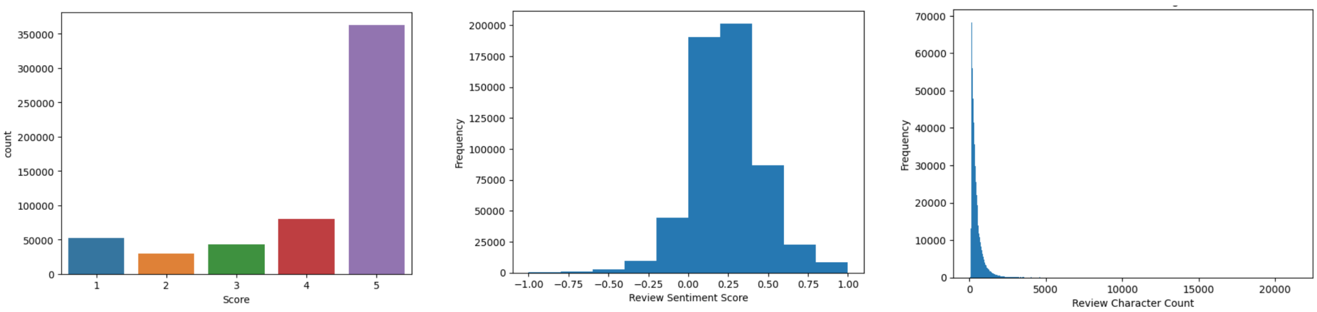
Memory and model-based approaches are two commonly used collaborative filtering based techniques. Memory-based (also called neighborhood-based) CF makes predictions based on the nearest neighborhoods. Neighborhood-based methods are further classified into two types: user-based methods and item-based methods. User-based CF will give a recommendation based on the user's past interaction history. On the other hand, item-based CF learns the relationships between items and recommends comparable items to a user based on their interaction history. In model-based CF, the latent factor model (LFM) is one of its variants, which captures the user and item's latent representations from high-dimensional data [3, 8].

CF recommender systems often compare group of users to other group of user based on a similarity measure though all these approaches often ignore the rich topics in the reviews given by certain users to certain products. These reviews can be vital in providing the personalized feedback to customers and enhancing their experience. As these reviews would contain the past interactions of certain users, will tell us that what sort of products does this particular user look for and what common themes or topics of liking or disliking does this user have.

We purpose a hybrid model that combines the Deep Neural Network with embedding based semantic representation for recommendations that would extract deep abstractions and non-linear feature representations [10]. Our approach, Neural Collaborative filtering (NCF) is truly collaborative in a sense that would utilize the topic embedding instead of individual embedding to recommend.

1.1. Data

We used amazon fine food reviews data-set available on Kaggle here. It had total of 568,454 reviews from around 256,059 users on 74,258 products. It included unique identifier for users, products and ratings and plain text review.



(a) Review Score Distribution: Majority of the reviews got high scores

(b) Review Sentiment Score Distribution: Most of the reviews has positive sentiment

(c) Review Length: Majority of reviews are less than 500 character

Figure 1: Exploratory Data Analysis

2. Approach

Figure 2 provides a comprehensive visual representation of the recommendation system's end-to-end process. The subsequent section offers an in-depth explanation of this depicted workflow.

2.1. Data Pre-processing

Since our approach included using the textual data to extract semantic features that would be essentially inputs to our deep learning framework for recommendations. There were multiple data pre-processing steps that are as follows:

- We cleaned our reviews data by removing all non alphabetic characters and stop words as they would not add any value to the semantic representation of the review.
- Since we were doing topic modeling hence we made sure that sentences with just one words are nouns this ensured more meaningful topics.
- We performed stemming for text normalization and reduced all the words to their root as this would help us consolidate the words with same base and enhance the topic modeling.

After cleaning our text data we analyzed the it using word cloud to understand it better and to ensure that we have the data in the format we want it to be.

Figure 3 indicates that the most frequently appearing terms include phrases such as ”highli recommended,” ”groceri store,” ”dog food,” ”cat food,” and ”dog love,” among others. The presence of recommendations for specific products suggests that other users with similar preferences may also appreciate these items.

2.2. Topic Modeling

As with most recommendation models the first step usually includes matrix factorization to create two low rank matrices, one for users and one for product from user-product interaction matrix. We had around 256,059 users and 74,258 products and their interaction matrix was of size 1.9×10^{10}. it was very difficult for us to work with this matrix due to its size, it was computationally intensive. We utilized topic modeling to cluster our users and products so that we can reduce the size of this interaction matrix and efficiently work with it.

Since we were clustering users and products we firstly created a history of users and products and find the all the reviews against each. We did this so that we can correctly identify a user and a product in semantic space based on all the interactions that included them.

Using this history we performed topic modeling to cluster the user and products. Steps included to convert the textual data into an embedding space, then performing dimensionality reduction and clustering and lastly fine tuning and evaluating the topics extract. Our topic modelling approach is similar to what is described here [5] with just additional step of grouping users and products to create their respective histories. Details can be found below.

2.2.1 Embeddings

we embed reviews of users and products to create representations in vector space that can be compared semantically. We assume that reviews containing the same topic are semantically similar [5]. To convert our documents into embedding space we utilized state-of-the-art pre-trained sentence transformer embedding model ***distilroberta***. This model is a distilled version of the RoBERTa-base model details can be found here. After passing our reviews through this model we were able to get 768 dimensional dense vector representation of every user and product.

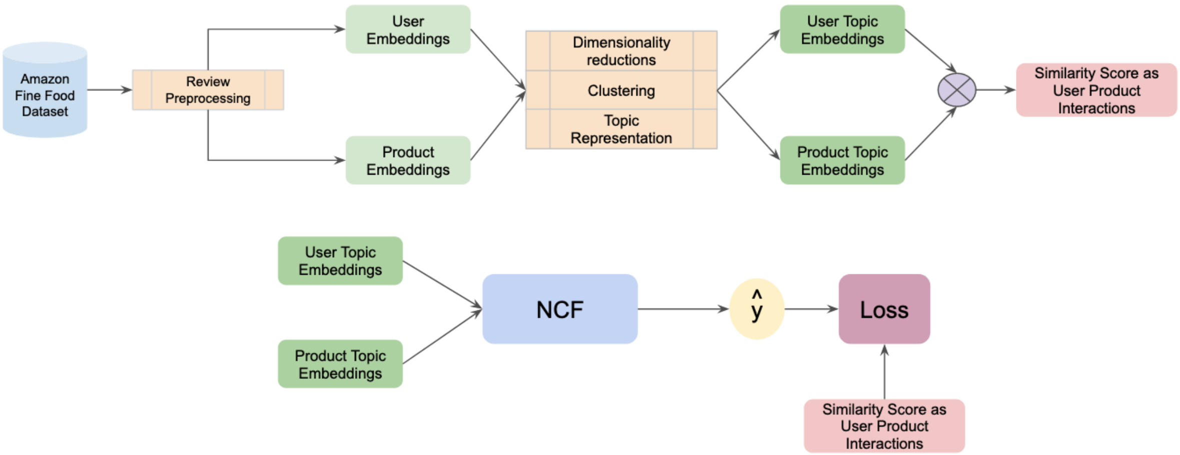


Figure 2: E2E Flow of recommendation system using semantic representation of User and Product, and NCF

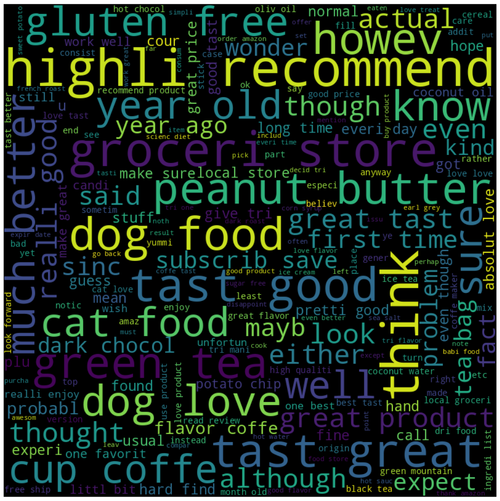


Figure 3: Word Cloud

2.2.2 Dimensionality Reduction

As clustering can become difficult due to curse of dimensionality hence to cater for it we performed dimensionality reduction using UMAP. We utilized UMAP since it has several advantages such as can capture both the local and global high-dimensional space in lower dimensions, arguably preserves more of the global structure with superior run time performance, able to scale to significantly larger data set and UMAP has no computational restrictions on embedding dimension, making it viable as a general purpose dimension reduction technique for machine learning [7].

2.2.3 Clustering

After performing dimensionality reduction we clustered the reduced representations using Kmeans. We utilized Kmeans as we wanted hard clustering rather than soft clustering and as [5] suggested that reducing high dimensional embeddings with UMAP can improve the performance of well-known clustering algorithms, such as k-Means and HDBSCAN, both in terms of clustering accuracy and time.

To find the topics we had to group similar representations together and hence we found 12K clusters for users and 10K clusters for products. Each of these clusters can be thought of a topic so essentially we found 12k topics for users and 10k topics for products.

2.2.4 Topic Representation

After extracting the topics we need some representation of them we can do this using count vectorizer that would essentially find the count of every token in a cluster and then using this bag of words representation we used class/topic based TF-IDF to find the important terms in a particular topic. We generalize TF-IDF procedure to clusters of documents. First, we treat all documents in a cluster as a single document by simply concatenating the documents. Then, TF-IDF is adjusted to account for this representation by translating documents to clusters:

$$W_{t,c} = t_{f,c}.\log(1 + \frac{A}{tf_t}) \tag{1}$$

Where the term frequency models the frequency of term t in a class c or in this instance. Here, the class c is the collection of documents concatenated into a single document for each cluster. Then, the inverse document frequency is replaced by the inverse class frequency to measure how much information a term provides to a class. It is calculated by taking the logarithm of the average number of words per class A divided by the frequency of term t across all classes. To output only positive values, we add one to the division within the logarithm [5].

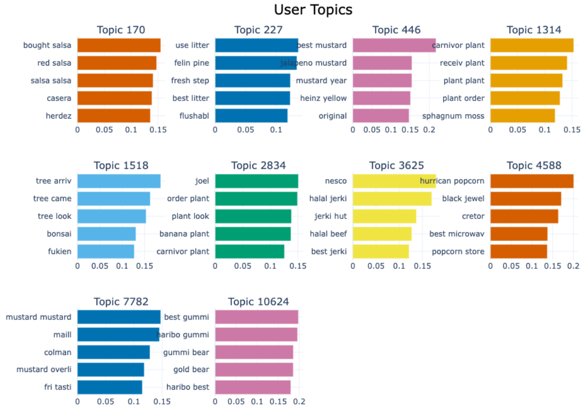


Figure 4: User Topics

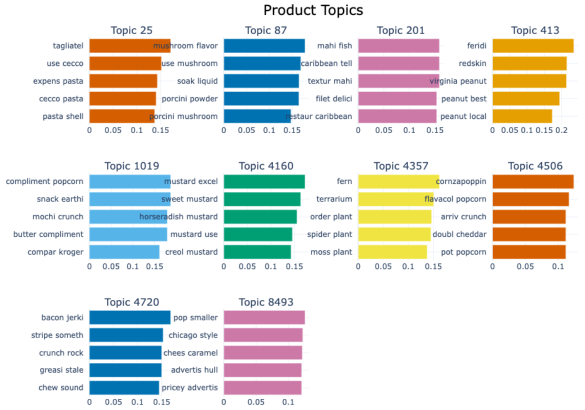


Figure 5: Product Topics

We can see major topics and their keywords in figure 4 for users and figure 5 for products and visually inspect them.

2.2.5 Fine Tuning

After finding the topics we can fine tune them to further enhance them. When we calculate the weights of keywords, we typically do not consider whether we already have similar keywords in our topic. Words like ”bikes” and ”bike” essentially represent the same information and often redundant.

To decrease this redundancy and improve the diversity of keywords, we can use an algorithm called Maximal Marginal Relevance (MMR). MMR considers the similarity of keywords/keyphrases with the document, along with the similarity of already selected keywords and keyphrases. This results in a selection of keywords that maximize their within diversity with respect to the document [5].

2.3. Neural Collaborative Filtering

Topic modeling has enabled us to capture semantic representations of products and users based on reviews, as well as to assess the similarity between them. The Neural Collaborative Filtering (NCF) [10] model is crafted to understand the function that represents user-item interactions, such as similarity, which can then be applied to forecast the probability of a user engaging with an item, for instance, a user providing a rating for a product. This model processes the unique identifiers of a user and an item to generate a predictive score for their potential interaction.

Note: Product and Item terms are interchangeably used.

2.3.1 User Topic Matrix and Product Topic Matrix

We have a user topic matrix with dimensions of 12,000 by 768 and a product topic matrix sized at 10,000 by 768. These matrices were formed by first translating the aggregate of review texts into a semantic space, then finding the specific topics associated with each user and product. Subsequently, we averaged their embeddings to construct the matrices that represent the topics related to users and products, as thoroughly explained in section 2.2. These matrices serve as pre-trained embeddings in the NCF model.

2.3.2 Model Architecture

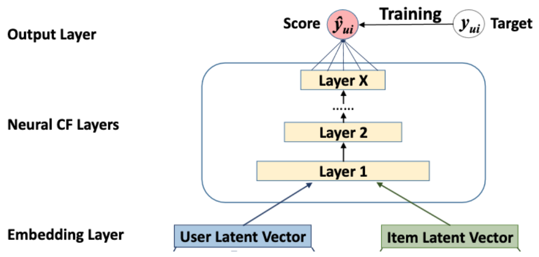


Figure 6: NCF model architecture: arXiv:1708.05031v2

The proposed NCF model is designed to capture the latent interactions between users and items. The model accepts as input the identifiers of a user and an item, and outputs a prediction representing the likelihood of interaction between the user and the item. The architecture of the model is as follows:

- Embedding Layers: The user and item identifiers are independently passed through distinct embedding layers. These layers transform the identifiers into dense vector representations, effectively capturing the latent factors associated with each user and item. The embedding layers are initialized with pre-existing embeddings and are held constant during training.

- Element-wise Multiplication: The resulting user and item embeddings are subjected to an element-wise multiplication operation, yielding an interaction vector. This operation can be mathematically represented as:

$$\text{mul} = \text{user\_embedding} \odot \text{item\_embedding} \tag{2}$$

- Concatenation: The user embeddings, item embeddings, and interaction vector are concatenated together to form a unified vector.

- Fully Connected Layers: The concatenated vector is subsequently passed through a series of fully connected layers, each equipped with a rectified linear unit (ReLU) activation function. Each layer can be mathematically represented as: $dense = ReLU(W * previous\_layer + b)$, where W and b denote the weights and biases of the layer, respectively.

- Output Layer: The final fully connected layer is linked to a single output neuron without an activation function. This neuron produces the predicted interaction, which can be mathematically represented as: $prediction = W * last\_dense\_layer + b_i$.

- Loss Function: The model employs the mean squared error loss function, which is appropriate for regression tasks. The loss function can be mathematically represented as: $loss = mean((prediction - true\_interaction)^2)$.

3. Experiments and Results

Before evaluating and presenting results for our NCF model we evaluated our topic model using the framework called OCTIS as presented here [9]. In addition to visually inspecting the topics we also used diversity metric to evaluate the topics for separately for users and for products. The diversity score is the metric that measures how diverse the top$k$ words of a topic are to each other. We chose $k = 10$ and diversity score for users topics was **0.99** while for products it was **0.97** both suggest that both models were good while users model being slightly better. Once we were confident with our topic models we went on with NCF model.

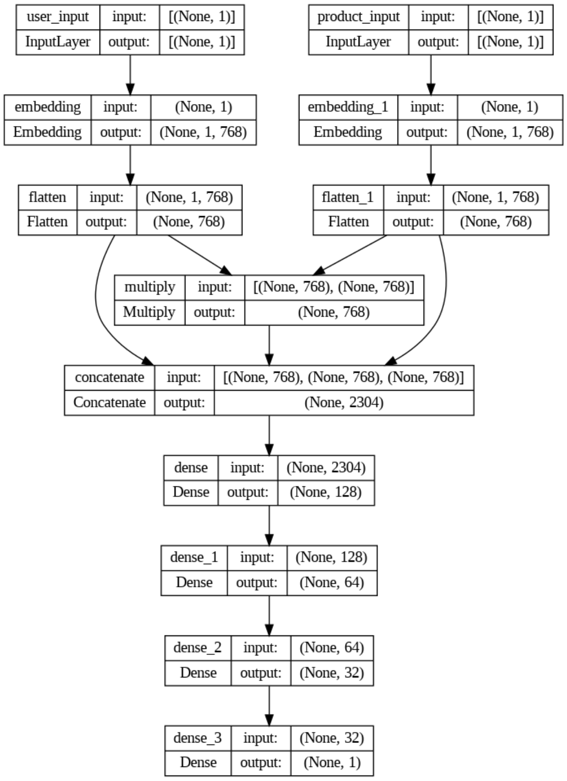

user_input | input: | [(None, 1)]
InputLayer | output: | [(None, 1)]
product_input | input: | [(None, 1)]
InputLayer | output: | [(None, 1)]
embedding | input: | (None, 1)
Embedding | output: | (None, 1, 768)
embedding_1 | input: | (None, 1)
Embedding | output: | (None, 1, 768)
flatten | input: | (None, 1, 768)
Flatten | output: | (None, 768)
flatten_1 | input: | (None, 1, 768)
Flatten | output: | (None, 768)
multiply | input: | [(None, 768), (None, 768)]
Multiply | output: | (None, 768)
concatenate | input: | [(None, 768), (None, 768), (None, 768)]
Concatenate | output: | (None, 2304)
dense | input: | (None, 2304)
Dense | output: | (None, 128)
dense_1 | input: | (None, 128)
Dense | output: | (None, 64)
dense_2 | input: | (None, 64)
Dense | output: | (None, 32)
dense_3 | input: | (None, 32)
Dense | output: | (None, 1)

Figure 7: NCF Model summary

Moreover, in the course of our experiments, we trained the Neural Collaborative Filtering model under varying degrees of complexity. The most complex variant of the model incorporated four dense layers, weight decay, and dropout regularization succeeding each layer. On the other hand, the least complex model was composed of merely two dense layers and did not include any form of regularization.

Interestingly, a model of intermediate complexity as shown Fig 6, comprising three dense layers, yielded the most promising results. The optimal hyperparameters for this model were determined to be a learning rate of 0.001, a batch size of 1024, and a total of 5 training epochs.

The performance of this model was evaluated based on the training, validation, and test losses. The model achieved a training loss of 1.2275e-06, indicative of its ability to learn effectively from the training data. The validation loss was recorded as 2.1576e-08, suggesting that the model was able to generalize well to unseen data. Finally, the model demonstrated a test loss of 2.1152935e-08, further attesting to its robust predictive performance on entirely new data.

In our analysis, we observed that users who posted reviews about soups were clustered into a single topic. A selection of these user reviews is presented in Table 1. Based on their interest in soups, these users were recommended products related to this category, as depicted in Table 2. Notably, these recommendations included items such as soya sauce and other Japanese-related products. This aligns with the understanding that soup is a significant component of Japanese cuisine. Thus, the recommendation system effectively identified and catered to the users' apparent interest in soup-based dishes, particularly those of Japanese origin.

Table 1: Users who reviewed about soups

UserId	Review	Topics
A106FKR29LW62P	favorit soup sold store live buy amazon best	64
AZS24SXHP3R1E	sever brand soup market groceri store similar tri sever disappoint point throw remain product soup distribut edward son trade carpinteria ca use higher qualiti ingredi other tast lot use broth cook rice	64

4. Conclusion

In conclusion, this project has demonstrated the effectiveness of utilizing Neural Collaborative Filtering (NCF) for product recommendation, as an alternative to traditional matrix factorization methods. A key innovation of our approach lies in the incorporation of semantic representations of users and products, which were employed as pre-trained embeddings in the model. By leveraging these semantic representations, our model was able to capture more nuanced user-product interactions, going beyond mere transactional data to encompass the underlying characteristics and preferences of users and products. This resulted in a more sophisticated and potentially more accurate recommendation system.

Looking ahead, there are several avenues for further enhancing the performance and utility of our model. One potential improvement could involve refining the semantic representations of users and products, perhaps by incorporating additional sources of data or employing more advanced natural language processing techniques. Additionally, the model's performance could potentially be boosted by exploring more complex architectures or more sophisticated training strategies. For instance, techniques such as learning rate scheduling, advanced regularization methods, or ensemble methods could be investigated.

References

[1] J. Konstan B. Sarwar, G. Karypis and J. Riedl. Item-based collaborative filtering recommendation algorithms, 2001. 1

[2] Georgios Demirtsoglou Stefanos Antaris Charilaos Zisopoulos, Savvas Karagiannidis. Content-based recommendation systems, 2008. 1

[3] Fethi Fkih. Similarity measures for collaborative filtering-based recommender systems: Review and experimental comparison. *Journal of King Saud University - Computer and Information Sciences*, 34(9):7645–7669, 2022. 1

[4] G Geetha. A hybrid approach using collaborative filtering and content based filtering for recommender system, 2018. 1

[5] Maarten Grootendorst. Bertopic: Neural topic modeling with a class-based tf-idf procedure, 2022. 2, 3, 4

[6] W. Liu X. He H. Luan H. Zhang, F. Shen and T.-S. Chua. Discrete collaborative filtering, 2016. 1

[7] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020. 3

[8] Qiao; Gan Yanglei; Dai Tingting; Leka Habte; Ayenew-Melak Tegene, Abebe; Liu. Deep learning and embedding based latent factor model for collaborative recommender systems. *Applied Sciences*, 13(2):726, 2023. 1

[9] Silvia Terragni, Elisabetta Fersini, Bruno Giovanni Galuzzi, Pietro Tropeano, and Antonio Candelieri. OCTIS: Comparing and optimizing topic models is simple! In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 263–270. Association for Computational Linguistics, Apr. 2021. 5

[10] Hanwang Zhang Liqiang Nie Xia Hu Tat-Seng Chua Xiangnan He, Lizi Liao. Neural collaborative filtering, 2017. 1, 4

Table 2: Products being recommended

ProductId	**Review**	**Topics**
B0000CNU5A	use think soya sauc pretti much salti heavier saltier other light less salti understand soya sauc happen get small dispos sampl yamasa soya sauc via takeout retail never bother use one day ran soya sauc put nishiki premium sushi rice along drop sesam seed oil wow flavour top road around bend noth like soya sauc ever tri total differ wonder flavour light salt work harmoni sesam seed oil sushi rice honestli need ingredi like chip peanut wo stop done man soya sauc great buy bottl tri leav chines version shelf like pearl river superior light soya sauc yamasa take soya sauc whole new level go good japanes restaur chanc yamasa tabl ca better product remind thousand lunch chicken teriyaki mound steam rice eat drizzl soy food home make everyday teriyaki day	71
B0000WKU2Q	happi see sauc amazon buy soon current bottl run unagi kabayaki sauc like use sushi restaur put unagi spider roll exact flavor consist could say similar teriyaki sauc realli less sweet stronger flavor thinner textur realli thing delici love use broil salmon catfish well highli recommend seafood lover anyon seek amaz flavor home	71
B00011EXP6	got local japanes food store first time great everi time worth tri probabl everyon spici	71

Student Name	Contributed Aspects	Details
Syed Mehlial Hassan Kazmi	Data Preprocessing and BerTopic	Performed data preprocessing for inputs to BerTopic, ran topics models and provided inputs for subsequent steps
Shilpa G	EDA and Implemented NCF	Performed exploratory data analysis. Using inputs from topic models trained and evaluated the best NCF models for recommendations

Table 3: Contributions of team members.